Deep Learning (CSE641/ECE555)

Assignment 3 (5 Marks)

Generative Adversarial Text-to-Image Synthesis

In this assignment, you will learn about text-to-image synthesis using conditional GANs. A typical GAN has a Generator (G) that takes random noise as input to generate realistic data samples (e.g., images or audio or text) and a Discriminator (D) that acts as a binary classifier, distinguishing between real and generated data. In Conditional GANs, input to (G) is conditioned over additional information.

In this assignment, you have to train a conditional GAN to generate images where input to **Target Generator** (G) is conditioned over textual descriptions. In addition, you have to train a **Source Encoder**, which will provide learned representations as input to (G) instead of noise. You may train the whole setup in an end-to-end manner or in parts. For instance, one approach could be knowledge distillation from source encoder to generator.

Overall Setup:

1. **Source Encoder:** Takes input image and outputs a representation. Any model size or type.
2. **Target Generator:** Takes representations from the source model and text encoding to generate new samples. The number of parameters should be half of that of the Source Encoder. Any model type.
3. **Discriminator:** Distinguishes between real and generated data. Any model size or type.

Rules:

1. You can use any library to design your GAN.
2. You can use any loss function, coding style, batch size, optimizer or learning rate scheduler.
3. You can use any model architecture except modern ones, such as transformer or diffusion-based models. (If you are unsure, please ask & clarify first.)
4. You can use the following as base repo for data: `https://github.com/aelnouby/Text-to-Image-Synthesis?tab=readme-ov-file`
5. You cannot use any pretrained model/checkpoint, i.e., all parameters in your setup should be trained from scratch (some random seed).
6. You have to demonstrate your setup by randomly selecting 20 classes (for the train) and 5 classes (for the test) from the Oxford-102 dataset. Text descriptions are available in the GitHub repo mentioned above.
7. Source encoder can not use class labels during training. You may use any loss function to make it as discriminative as possible for the real images of all 25 classes.
8. We will only run & test your code on Google Colab. You have a maximum of 200 epochs for training using Colab resources. Time per epoch doesn't matter but it is advisable that the training and testing can be finished within 1 hr (though not mandatory). Hence, choose a resonable model size.
9. We encourage you to save .ipynb file cell outputs such as plots, visualization, loss/acc logs etc to aid in subjective evaluation component.

```
[Input] --> [Source Encoder] --> [Representation] --> [Target Generator] --> [Generated Image]
                                                                ^
                                                                |
[Text Input] --> [Text Encoder] --> [Text Encoding] ------------+

[Generated Image] --> [Discriminator] <--> [Real Images]
```

Deliverables:

1. We don't need your trained model but a robust code that can replicate your best setting.
2. Submit a single .ipynb file for this assignment with clean documented code. Beautifully structure your notebook as if you are given a demo tutorial to a 1st year B.Tech student who can easily follow the steps.
3. Highlight the innovations (new things), if any, you have used that you believe make your submission stand out and different from the entire class.
4. There should be two separate sections, one for Training and one for Testing.
5. In Training/Testing, you may use the dataloader from the above-mentioned GitHub repo.
6. In Testing, using the best model checkpoint you have to
   (a) Generate and plot 5 random images from each test class as a grid of 5x5. (Hint: use diverse unseen text.)
   (b) Plot the 3D-tSNE embedding of Source Encoder on all images from both train and test sets.
   (c) Print in the form of a table: the total number of parameters, number of trainable parameters and model size on disk for encoder, generator and discriminator.

Marking:

This assignment will not be fully auto-graded. Marking will be manual with subjective evaluations using the following components:

1. Overall structure & cleanliness of submitted code notebook [1 mark]
2. Successful training of the full GAN model [1 mark]
3. Discriminative ability of the embeddings from Source encoder [1 mark]
4. Subjective diversity and quality of generation [1 mark]
5. Subjective evaluation of innovation in model architecture (including its size and memory footprint) and training paradigm [1 mark]